GA–A25245

DATA GRIDS FOR LARGE-SCALE FUSION SIMULATIONS

Final Report for the period February 18, 2004 through July 29, 2005

by

PROJECT STAFF

JANUARY 2006

GENERAL ATOMICS

GA–A25245

DATA GRIDS FOR LARGE-SCALE FUSION SIMULATIONS

Final Report for the period February 18, 2004 through July 29, 2005

by
PROJECT STAFF

Prepared for
Tech-X Corporation
under U.S. Department of Energy
Contract No. DE-FG02-03ER83842

GENERAL ATOMICS PROJECT 30215
JANUARY 2006

GENERAL ATOMICS

1. INTRODUCTION

Fusion experiments and simulations generate large data sets. For example, the DIII-D experiment produces multiple Gigabytes of experimental data for a single 6-second pulse of the machine. Over time the size and number of data sets has increased and can be expected to continue to increase (Fig. 1). These larger data sets bring new challenges to data management in fusion research. Of particular concern is the difficulty in managing simulation data. To begin with, simulation data is very large, sometimes in the hundreds of Gigabytes—even Terabytes—for a single code run. In addition, it is hard to plan for the data storage needs of simulations because it is not possible to predict how many simulation runs will be done. This is in contrast to the data storage needs of fusion experiments where, given knowledge of the number of Gigabytes per shot, the number of shots per day, and the number of experimental days per campaign, one can roughly predict data storage needs many months in advance.

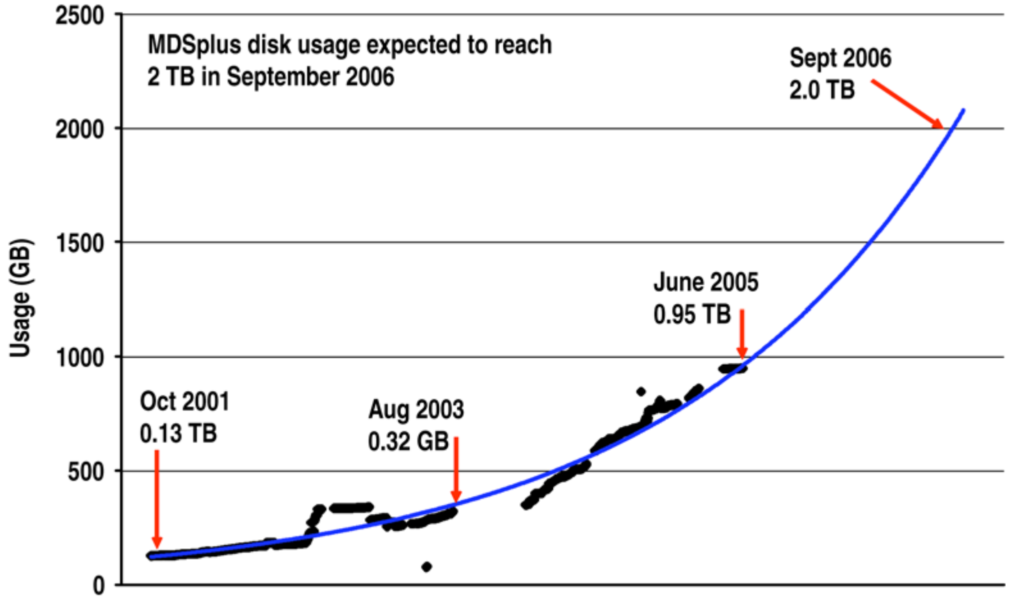

Fig. 1. MDSplus storage needs at DIII-D are increasing exponentially

This contrast between simulation and experimental datasets is illustrated in the effect of the NIMROD simulation code on the daily change in MDSplus disk usage requirements at DIII-D (Fig. 2). During normal operations, disk usage changed much less than 1% per day. However, during the period of time when NIMROD output was written to MDSplus, disk usage could change as much as 20-25% in a single day.

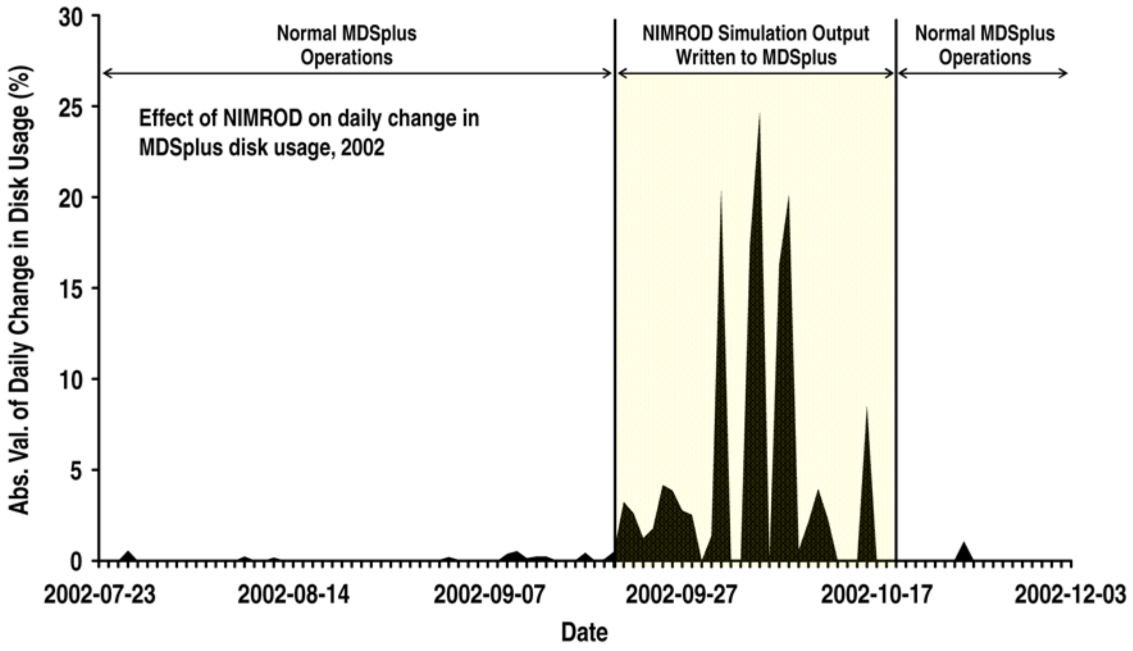

Fig. 2. Simulation datasets such as NIMROD are very large and result in large daily changes in disk usage required for MDSplus data.

One approach to dealing with huge fluctuations is to simply purchase enough data storage capacity to meet the possible needs. This is, however, not economical since a 20% daily increase translates into a 100-fold increase in storage needs every quarter. Moreover, preparing for such a need would in many cases result in a large amount of excess capacity. What is needed is instead a system where storage can be quickly scaled as needed.

It was with this need in mind that researchers investigated the use of distributed client MDSplus. MDSplus is a data acquisition and management system used by 30 fusion sites worldwide. As a data storage system, MDSplus supports local and remote access, host-based or certificate-based authentication, and data compression. The new “distributed client” capability distributes expression evaluation—the heart of MDSplus—to data clients rather than servers. Moreover, the use of distributed MDSplus makes it possible to move from a monolithic architecture (Fig. 3) to one where data is distributed onto a “virtual server” consisting of multiple servers (Fig. 4).

A distributed server architecture is more scalable than a monolithic architecture for several reasons. Most obvious is the benefit that new servers may be added to the virtual server as needs increase. For example, one can add more storage capacity to existing hosts if more storage is needed or more servers if more bandwidth is required. Moreover, adding or removing hosts need not result in service interruptions provided that sufficient virtual server hosts remain online to support user needs. This stands in contrast with a monolithic system where adding new hardware typically requires complete data unavailability.

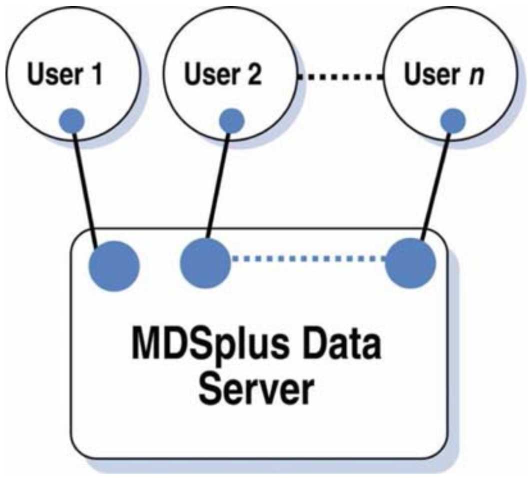

Fig. 3. In a traditional monolithic architecture using thin client MDSplus, most of the work is done server-side, and each client connects to the same server.

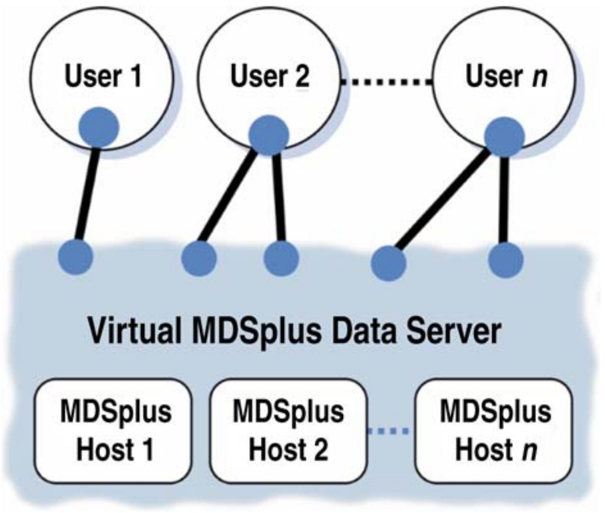

Fig. 4. A distributed architecture allows for the creation of a virtual data server and results in the client sharing much of the load of the system.

2. TOWARDS SCALABLE MDSplus STORAGE

Developers hypothesized that the new distributed client MDSplus could be used in such a way to improve upon the present thin client technology by distributing load onto multiple hosts. It was also hypothesized that, since adoption of distributed client MDSplus would eliminate the extra network hop for data transfer when storing PTDATA references in MDSplus trees, that the data transfer rate would be improved because of more efficient use of network resources.

Developers first designed a set of tests to investigate these ideas. It was determined to focus on data serving capability since most of the time data is being read from the server, not written to the server. It was decided to compare distributed and thin client MDSplus at different load levels, with PTDATA references, with node reference indirection, with node math expressions, and with compression turned off. It was also decided to perform two systems tests—one to test the effect of TCP window adjustment, and another to test the effect of storing MDSplus trees in a RAM disk as opposed to hard disk. Developers selected a cluster of Linux hosts—each host being identical and using the same network—to serve as MDSplus clients. Three workstation-class Linux machines were identified for use as data servers.

A suite of test scripts were created to implement the performance tests. The test scripts record performance measurements to flat files, which are subsequently imported into a relational database for analysis.

For testing, 12 Linux clients and 3 Linux servers were configured as required. MDSplus and the test scripts were put onto the clients. The servers were configured to serve MDSplus data stored on local disk (and RAM disk) through the standard xinetd mechanism.

As hypothesized, it was found that the use of distributed client MDSplus did increase data transfer rates when using PTDATA references in nodes. In this particular case the performance increase was 15%.

A very interesting result was that, while thin client and distributed client perform more or less the same at low loads, under higher loads the distributed client performance gain is very large. This is because thin client puts most of the work on the server, and very little work on the client. As system load increases under thin client, the server eventually becomes a bottleneck as it is doing all of the work. In contrast, under distributed client most of the work is done client-side, thus keeping the load down on the data server. In this particular test the performance advantage was 20% under moderate loads.

Tests also revealed that under all load conditions distributed MDSplus makes better use of network bandwidth because it sends data over the wire using compression. With older, slower computers it might be the case that the extra computational load required for compression and decompression was too much of a tradeoff for the bandwidth conserved (for LAN transfers). However, it seems that with the modern computers used in this round of testing, that the tradeoffs definitely favor the use of compression. Turning compression off resulted in a 45% decrease in performance for distributed MDSplus under low loads.

The systems adjustments performed—adjusting TCP window size and storing MDSplus data on RAM disk—did not improve performance. In fact, tests show that moving MDSplus data onto RAM disk actually degraded performance, presumably because the reduction in RAM available to the OS more than offset any IO gains made by storing a file in memory as opposed to hard disk.

3. FOLLOW-UP WORK

Results of the MDSplus client comparison tests are described in the paper “MDSplus Testing at DIII-D”, authored by David Minor and Justin Burruss, and submitted for publication in *Fusion Engineering and Design*. A poster was presented by Mr. Minor at the IAEA-TM on Control, Data Acquisition, and Remote Participation for Fusion Research in July 2005.

More comparison tests were conducted by Qian Peng at General Atomics. Ms. Peng made use of the same test scripts used in the original set of tests to measure data transfer rate at a broader range of loads (Fig. 5). The results support one of the conclusions of the original set of tests: distributed client outperforms thin client under higher loads.

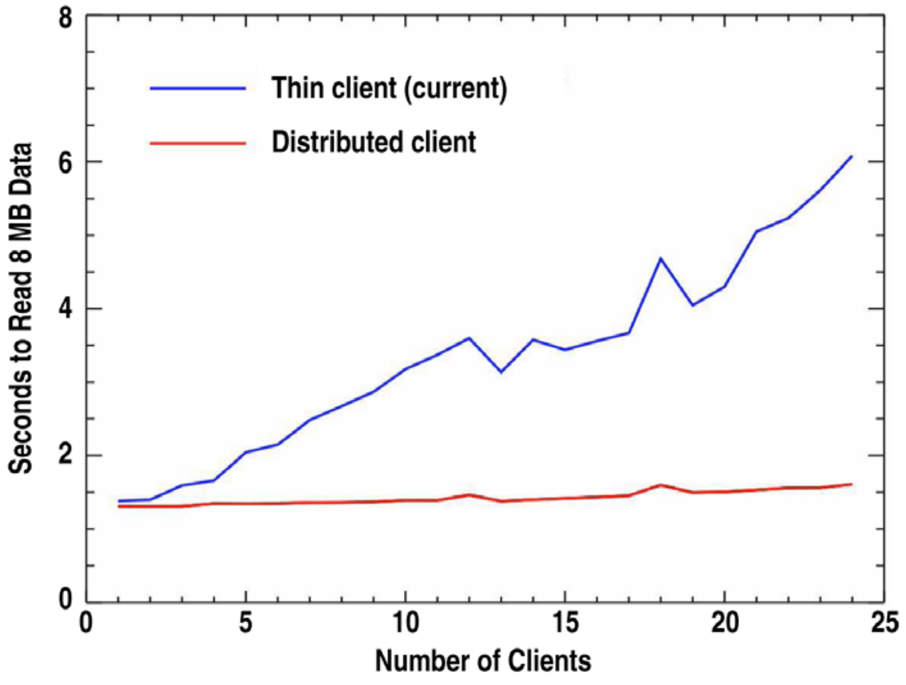

Fig. 5. Distributed client MDSplus scales better than thin client MDSplus.

4. NEXT STEPS

Data Grids test results are presently being used by Justin Burruss of General Atomics to design and implement a next-generation MDSplus system architecture to keep pace with the increasing demands fusion experiment and simulation. This task will require software updates and extensive testing. To find the optimal design will also require more fine-grained resource monitoring: individual trees accesses, not just server load, will require usage monitoring. In the short-term this work is to be done for DIII-D, but in the longer-term lessons learned may be used by other fusion facilities using MDSplus to improve data throughput for experimental and simulation data. For simulation data, one can imagine a scalable fusion data grid where, when more capacity is required, new machines are simply added to the grid. The International Thermonuclear Experimental Reactor (ITER) will most certainly require some kind of scalable data storage solution during its planned 20 years of operation and beyond.

ACKNOWLEDGMENT

This report of work was prepared for Tech-X Corporation under U.S. Department of Energy Contract No. DE-FG02-03ER83842.